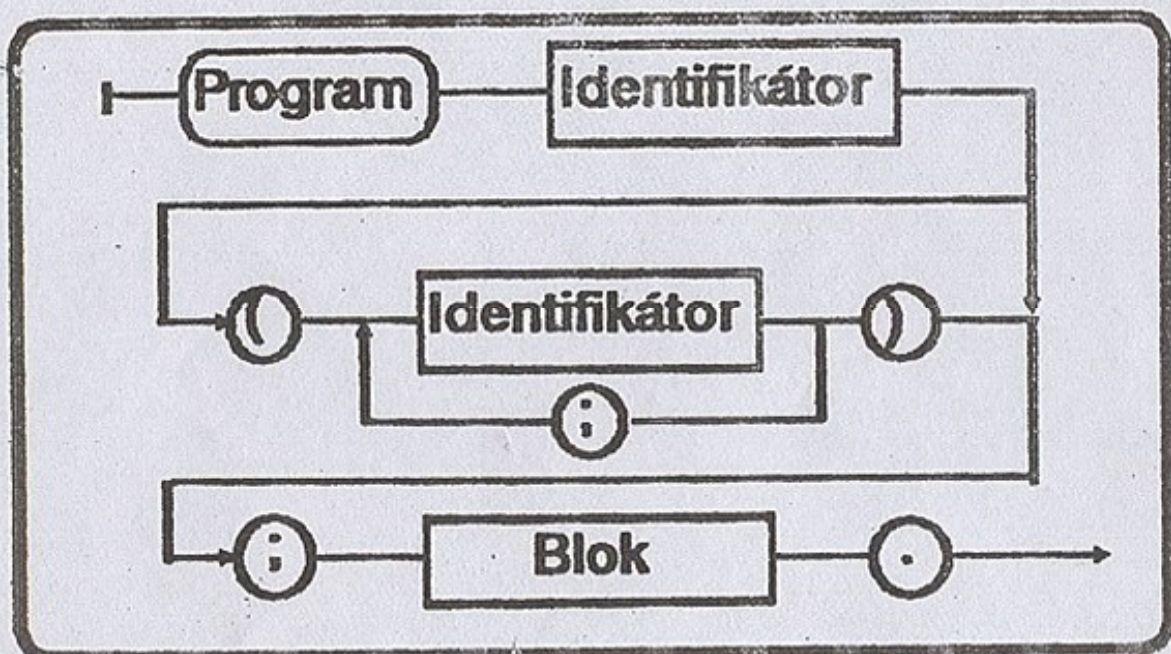


PŘÍRUČKA PROGRAMÁTORA

Pascal C-2717 V2.C



Mikropočítač Consul 2717 "Zbrojováček"



INCOTEX
Brno, Hybešova 42



Stanice mladých techniků
Brno III, Holubova 18

Pascal C-2717 je překladač podmnožiny programovacího jazyka Pascal. Je vhodný na výuku programování na středních školách a v zájmových útvarech základních škol, Domů dětí a mládeže, Stanic mladých techniků.

1. Spuštění překladače

Překladač pracuje pod operačním systémem CP/M. K jeho práci je potřebná jen přítomnost souboru PASCAL.COM na disketě. Překladač spustíme, když na odezvu systému (například A) napíšeme PASCAL a stiskneme EOL.

2. Hlavní menu

Po spuštění překladače se objeví hlavní menu. Obsahuje všechny základní volby, kterými ovládáme práci systému. Volbu z menu zvolíme stlačením prvního písmene v názvu volby.

Editor	Vyvolání editoru.
Překlad	Překlad programu.
Spuštění	Spuštění programu (když byl od posledního překladu vyvolán editor, tak se nejprve program přeloží). Program je možné během práce zastavit zmáčknutím přemyku a WRK.
Konec	Ukončení práce, návrat do CP/M. Pokud jsme neuložili program na disk, ztratí se.
Nový	Smazání programu a jeho jména.
Zápis	Uložení programu na disk pod jménem, které je uvedené v menu (když program nemá jméno, počítač si ho vyžádá, s prázdným jménem nemůžeme program nahrát na disk). Vytvořený soubor na disku je běžný textový soubor.
Čtení	Načtení programu z disku. Počítač si vyžádá jméno souboru. Pro jeho zadání platí běžné konvence operačního systému CP/M. Když neuvedeme příponu, dosadí se .PAS. Program se připojí za program v paměti. Tedy když nechceme, aby byl program připojený za existující, použijeme před čtením volbu Nový, která zruší program v paměti.

Tisk	Tisk textu na tiskárně.
Jméno	Přejmenování programu v paměti.
Index	Zapnutí a vypnutí kontroly indexů. Když je za ním +, bude se při každém indexování pole kontrolovat, zda je index v rozsahu podle deklarace pole. Když je za ním -, kontroly se nevykonávají - program poběží rychleji, ale v případě nesprávného indexování může narušit systém.

3. Editor

Do editoru se vchází z hlavního menu stisknutím písmene E. Editor se ukončí současným stisknutím přemyku a WRK.

Editor je obrazovkový: když stiskneme klávesu s písmenem (nebo jiným tisknutelným znakem), napíše se do řádku na pozici kurzoru. Délka řádků může být maximálně 127 znaků, ale doporučuje se používat řádky do 64 znaků. Když píšeme za 64. pozicí na řádku, řádek se začne "rolovat".

Každý řádek, z kterého odejdeme kurzorem se zpracuje a vypíše se zpět na obrazovku v upraveném tvaru: všechna klíčová slova jsou zobrazena malými písmeny a všechny identifikátory velkými písmeny. Mezery jsou v řádku rozmístěny tak, aby správně oddělovaly jednotlivé prvky jazyka. Když vznikne v řádku chyba (například chybí apostrof, nebo špatný zápis reálné konstanty), není možné z řádku odejít.

3.1. Funkce kláves v editoru

Znakem ^ označujeme klávesu přemyku. Klávesy se "šipkami s čárkou" nazýváme šipka nahoru a šipka dolů.

šipka vlevo	posun o znak doleva
šipka vpravo	posun o znak doprava
šipka nahoru	posun o řádek nahoru
šipka dolů	posun o řádek dolů
EOL	přechod na začátek následujícího řádku
šikmá šipka	posune kurzor na začátek řádku
END	posune kurzor na konec řádku

CLR	smazání obsahu řádku (celý řádek se zaplní mezerami)
^CLR	smaže znaky na řádku vpravo od kurzoru
INS	vložení mezery na pozici kurzoru (zbytek řádku se posune doprava)
DEL	smazání znaku na místě kurzoru (zbytek řádku se posune doleva)
RCL	vyvolání řádku v takovém stavu, jaký byl, když jsme do něho přišli šipkou (tedy "vrácení" řádku v takovém stavu, jaký byl před jeho změnou)
^HOME	nastaví kurzor na první řádek programu
^END	nastaví kurzor na poslední řádek programu
^šipka dolů	vyroluje program o 20 řádků dolů, zobrazí tedy "další stránku" programu
^šipka nahoru	vyroluje program o 20 řádků nahoru, zobrazí tedy "předěšlou stránku" programu
^INS	před řádek s kurzorem vsune prázdný řádek
^DEL	zruší řádek s kurzorem
^RCL	před řádkem s kurzorem vloží poslední zrušený řádek pomocí ^DEL
C-D	privolá do vrchního řádku poslední zprávu (většinou zprávu o chybě)
^WRK	ukončení editování.

4. Chybovník

Chyby při překladu a při běhu se ohlašují tak, že se na místo chyby nastaví kurzor v editoru. Chyby při editování se hlásí, když chceme řádek opustit kurzorem, kurzor zůstane na svém místě (neukazuje přesné místo chyby).

Syntaktické chyby (hlásí se při překladu)

- 1: Na začátku programu musí být slovo PROGRAM
- 2: Očekával se identifikátor
- 3: Špatný začátek deklarace
- 4: Chybí rovnítko
- 5: Chybí středník
- 6: Chybí dvojtečka
- 7: Chybí prava závorka
- 8: Špatný začátek popisu typu
- 9: Chybí levá hranatá závorka

- 10: Chybí dvě tečky
- 11: Chybí pravá hranatá závorka
- 12: Chybí OF
- 13: Chybí znak přiřazení - nepochopený příkaz
- 14: Chybí END
- 15: Chybí DO
- 16: Chybí UNTIL
- 17: Chybí TO nebo DOWNT0
- 18: Chybí levá závorka
- 19: Chybí tečka za posledním END nebo je END navíc
- 20: Špatný výraz
- 21: Špatný začátek příkazu
- 22: Špatná konstanta
- 23: Chybí THEN
- 24: Očekávalo se celé číslo

Semantické chyby (hlásí se při překladu)

- 26: Identifikátor není identifikátorem položky záznamu
- 27: Proměnná není typu ukazatel
- 28: Moc vnořených WITH
- 29: Proměnná není typu RECORD
- 30: Chybí deklarace typu, které byly použity v tomto bloku, v deklaracích typu ukazatel (hlásí až na BEGIN příslušného bloku)
- 31: Identifikátor není identifikátorem konstanty
- 32: Unární plus nebo minus pro typ různý od INTEGER nebo REAL
- 33: Špatný typ indexů v deklaraci pole
- 34: Pole nemůže mít víc než 65536 byte
- 35: Identifikátor není identifikátorem typu nebo typ ještě nebyl definován
- 36: Dva formáty jsou povoleny jen pro výrazy typu REAL, za proměnou typu soubor nemůže následovat formát
- 37: Nevhodný typ hranice v deklaraci intervalu
- 38: Nekompatibilní typy hranic v deklaraci intervalu
- 39: V intervalu nemůže být spodní hranice větší než horní
- 40: Nedeklarovaný identifikátor
- 41: Identifikátor byl už v tomto bloku použit, nemůže být předefinován
- 42: Identifikátor byl už v tomto bloku deklarovaný
- 43: Výraz v příkazu CASE nebo položka v CASE části deklarace RECORD musí být skalárního typu
- 44: Nekompatibilní typ navěští v příkaze CASE nebo v části CASE v deklaraci RECORD
- 45:
- 46: ABSOLUTE je použito jinde než v hlavním programu

- 47: Vícenásobný FORWARD
- 48: V těle FORWARD procedury nebo funkce už není možné opakovat parametry a typ
- 49: Chybní deklarace procedur nebo funkcí, které byly zadeklarovány jako FORWARD (hlásí se na BEGIN)
- 50: Nekompatibilní nebo nesprávné typy v celočíselné nebo reálné binární operaci
- 51: Nekompatibilní typy v logické binární operaci
- 52: Hranice není kompatibilní s proměnou cyklu FOR
- 53: Špatný typ v NOT
- 54: Špatný identifikátor na levé straně přiřazení
- 55: Indexovat je možné jen pole
- 56: Nekompatibilní typ indexu
- 57: Takovéto typy je možné porovnávat jen na = nebo <>
- 58: Přiřazení hodnoty funkce mimo jejího těla
- 59: Identifikátor není identifikátorem proměnné nebo funkce
- 60: Nekompatibilní typy při přiřazení nebo proměnné, které obsahují soubory není možné přiřadit
- 61: Duplicitní jméno položky v RECORD (hlásí až za typem položky)
- 62: Očekává se výraz typu BOOLEAN
- 63: Objekty tohoto typu není možné použít v READ a WRITE
- 64: Formát v příkazu WRITE musí být typu INTEGER
- 65: Nekompatibilní typy v relačním vztahu
- 66: Identifikátor není procedura
- 67: Moc parametrů
- 68: Nekompatibilní typ parametru nebo hodnotový parametr nemůže obsahovat soubor
- 69: Za var parametr nelze posílat výraz
- 70: Málo parametrů
- 71: Identifikátor není funkce
- 72: Identifikátor není identifikátorem proměnné
- 73: Špatný identifikátor v FOR (musí to být lokální proměnná)
- 74: EXIT není v žádném cyklu
- 75: Interní chyba - pokus o špatné adresování

Chyby při editování, při běhu a překročení omezení
při překladač

- 77: Konec souboru (EOF)
- 78: Parameter SORT je zaporný
- 79: Parameter LN je nekladný
- 80: Parameter EXP je větší než 99.0
- 81: Špatné jméno souboru v ASSIGN
- 82: Soubor neexistuje (v RESET nebo REWRITE)
- 83: Moc velká délka v BPLLOT (obrázek by byl mimo obrazovku)
- 84: Moc velká šířka v BPLLOT (obrázek by byl mimo obrazovku)

- 85: Špatné souřadnice v BPL0T (mimo interval 0..47 resp. 0..255)
- 86:
- 87: Přetečení při reálné operaci, při TRUNC nebo ROUND
- 88: Index mimo interval
- 89: Dělení nulou
- 90: Dlouhý řádek (při editování). Touto chybou editor označí situaci, když je po úpravě řádek delší než 80 znaků i když původně byl napsaný na méně znaků (může se to stát, například když je v řádku víc reálných konstant v semilogaritmickém tvaru). Z textu už existuje jen 79 znaků - zbytek je ztracený. Na konec řádku umístí editor znak '!' (vykřičník). Ten způsobí, že není možné z řádku odejít bez jeho úpravy. (Pozn.: když píšete řádky do 48 znaků, tato chyba se prakticky nevyskytne).
- 91: Chybí apostrof (při editování)
- 92: Špatný zápis celého nebo reálného čísla (při editování i při běhu). Tuto chybu zahlásí i tehdy, když zadáváte moc velké nebo moc malé číslo (kromě celých čísel při editování - ty se nekontrolují), nebo když má mantisa reálného čísla více než 38 čísel (není účelné psát víc než 7 čísel mantisy - číslo nebude přesnější, právě naopak, může se projevit zaokrouhlovací chyba).
- 93: Málo paměti (při editování, překladu i při běhu)
- 94: Příliš složitý výraz (při překladu)
- 95: Víc než 4 úrovně vnoření procedur a funkcí (při překladu)
- 96: Víc než 255 identifikátorů (při editování)
- 97: Přetečení tabulek kompilátoru (při překladu)
- 98: Byl dosáhnut konec textu před logickým koncem programu (varování při překladu)
- 99: Přetečení zásobníku analyzátoru - moc vnoření procedur, příkazů a výrazů (při překladu)
- 00: V řádku je znak, který nemá v Pascalu význam (při editování)
- 100: Pokus o DISPOSE ukazatele obsahem NIL
- 101: Soubor není otevřen pro vstup (v READ, READLN, EOF, EOLN)
- 102: Soubor není otevřen pro výstup (v WRITE, WRITELN, PAGE)
- 103: Chyba při zápisu do souboru - disk je asi plný
- 104: Před RESET nebo REWRITE nebylo použito ASSIGN (tato chyba se vždy nevyhlásí, ale i přesto je nutné použít ASSIGN před RESET a REWRITE)
- 105: Zařízení, které se přiřadilo v ASSIGN není vstupní (hlásí RESET) nebo výstupní (hlásí REWRITE)
- 106: Není možné vytvořit nový soubor - adresář je plný
- 107: Soubor se nedaí uzavřít (hlásí CLOSE, chyba vznikne například, když se použije na neotevřený soubor)
- 108: Zařízení LST není možné použít v ASSIGN

5. Jazyk Pascal C-2717

Pascal C-2717 je podmnožinou standardního Pascalu (při jejím výběru jsme vycházeli z podmnožiny Pascal-S, která je použita v učebnici N. Wirtha Systematické programovanie, slovensky překlad vyšel v Alfa v r. 1981) a je rozšířený o některé standardní procedury a o prostředky pro programování na nižší úrovni. V této části popíšeme velmi stručně změny v jazyce proti standardnímu Pascalu (např. knihu L. Molnár: Programovanie v jazyku Pascal, Alfa a SNTL 1987) a podrobněji popíšeme všechny funkce a procedury, které jsou v Pascalu C-2717 standardní a také implementaci vstupních a výstupních příkazů a procedur. Předpokládáme, že čtenář je obeznámen se standardním Pascalem.

5.1. Omezení proti standardnímu Pascalu

Ze standardního Pascalu nejsou implementovány tyto vlastnosti:

Deklarace LABEL

Typ údajů množina a FILE OF typ (textové soubory jsou implementovány)

Zhuštěné typy údajů

Parametry typu procedura, funkce a konformní pole

Příkaz GOTO

Standardní procedury GET a PUT (podobně jako v Turbo Pascalu).

Slova READ, READLN, WRITE a WRITELN považuje překladač za rezervované na rozdíl od standardního Pascalu, kde jsou to identifikátory standardních funkcí (v Pascalu C-2717 potom hovoříme o příkazu READ místo procedury READ).

5.2 Rozšíření jazyka

Většina rozšíření je motivována lepším přístupem k programování na "nižší úrovni", tedy na práci s pamětí, s jednotlivými bity proměnných a podobně. Začátečníci mohou

následující odstavce vynechat a vrátit se k nim později.

Zápis číselných konstant v šestnáctkové soustavě

Číselné konstanty je možné zapisovat také v šestnáctkové soustavě, když se před číslo uvede znak \$ (dolar). Například zápis \$FF je ekvivalentní zápisu 255.

Zápis řetězcových konstant

Mimo běžného zápisu řetězcových konstant (např. 'ABCD') je možné použít také zápis pomocí kódů jednotlivých znaků, kde před každým z nich je uveden znak #. Například zápis #11#12#13 představuje konstantu typu ARRAY[1..3] OF CHAR, která má prvky s hodnotou CHR(11), druhý CHR(12) a třetí CHR(13). Oba způsoby zápisu je možné mít například takto 'AHOJ'#13'JANO'. V zápise nesmí být mezery mimo apostrofy. V kódech je možné použít také šestnáctkovou soustavu: 'AHOJ'#\$OD'JANO'. Tento zápis konstant využijeme jako parametr procedury BLOT.

Určení adresy proměnné

Proměnným hlavního programu možno předepsat adresu jejich umístění pomocí slova ABSOLUTE, za kterým následuje celočíselná konstanta (s výhodou využijeme šestnáctkovou soustavu). ABSOLUTE zapisujeme za typ proměnné. Takto je možné přistupovat k proměnným monitoru, nebo zapsat údaje do speciálních oblastí paměti (to, co se v Basicu dosahuje pomocí PEEK a POKE).

Příklad:

```
VAR BDOSADR:CHAR ABSOLUTE 6;
```

Operátor XOR

Tímto operátorem je označena operace "exclusive or", v matematice nazývaná nonekvivalence. Operátor můžeme aplikovat na operandy typu BOOLEAN nebo INTEGER.

Logické operace s čísly

Logické operátory AND, OR, XOR a NOT můžeme použít také s operandy typu INTEGER. V tomto případě mají význam operací s bity čísel a výsledek je typu INTEGER.

Příkazy BREAK, HALT a EXIT

V Pascalu C-2717 není implementovaný příkaz skoku. V nejnútnejších případech ho můžeme nahradit "mírnějšími" (ale hlavně jasnějšími) příkazy, které ukončí vykonávání přesně určené části programu.

Příkaz BREAK ukončí vykonávání nejvnitřnějšího cyklu, v kterém se nachází a pokračuje na prvním příkazu za tímto cyklem.

Příkaz EXIT ukončí vykonávání procedury nebo funkce, v které se nachází (taková obdoba Basicovského RETURN).
Příkaz HALT ukončí vykonávání celého programu.

Část OTHERWISE v příkazu CASE

Pro pohodlnější programování příkazů CASE je možné použít poslední možnost OTHERWISE. Příkaz za OTHERWISE se vykoná, když výraz za CASE nebude mít ani jednu hodnotu z dříve vyjmenovaných případů:

```
CASE I OF  
  1: J:=A+B;  
  2: J:=A*B;  
  OTHERWISE J:=0  
END;
```

5.3 Změny, které vyplývají z koncepce editoru

Editor analyzuje každý jeden řádek samostatně a převádí ho do vnitřního tvaru. Z toho vyplývá několik omezení na zápis programu. Na většinu z nich narazíme jen ve velmi speciálních případech.

- Řetězcové konstanty a komentáře nemohou přecházet z jednoho řádku na druhý.
- Celočíselné konstanty rozeznává editor bez znaménka, tedy v rozsahu od 0 do 65535, když napíšeme větší celé číslo, editor ho "usekne" do tohoto intervalu. (Používat čísla nad 32767 je vhodné jen ve speciálních případech při programování na nižší úrovni).
- Reálnou konstantu v semilogaritmickém tvaru editor vždy zobrazuje s mantisou na 6 míst s desetinnou tečkou za první číslicí (když napíšeme 51.2E3, editor zobrazí 5.12000E+04).
- Řetězcové a znakové konstanty, které byly zapsané pomocí znaku # zobrazí editor podle toho, zda jde o tisknutelné znaky nebo ne (když napíšeme #65#1#66, editor zobrazí 'A'#1'B').
- Když byla v zápisu řetězcové konstanty použita šestnáctková soustava i jen v jednom znaku, editor zobrazí všechny netisknutelné znaky řetězce v šestnáctkové soustavě (když napíšeme #128#65#\$FF, editor zobrazí #\$80'A'#\$FF).

5.4 Implementace standardních typů

Typ INTEGER je implementovaný v 16 bitech. Má rozsah od -32768 po 32767. Standardní konstanta MAXINT má hodnotu 32767. Operace s typem INTEGER nehlásí chybu při přetečení.

Typ REAL je implementovaný v 32 bitech s 24-bitovou mantisou a 8-bitovým exponentem. Rozsah absolutních hodnot čísel je přibližně od $1E-38$ po $1E37$. Vnitřní přesnost je přibližně sedm platných cifer, při výstupu se však zobrazuje nejvíce šest platných cifer. Operace a funkce s typem REAL hlásí chybu při přetečení. Při podtečení je výsledek nula.

Typ CHAR je implementovaný v 1 byte, kód je podle tabulky ASCII s rozšířením pro znaky české a slovenské abecedy podle kódu KOI-8-čs-2. Rozsah kódů znaků je od 0 do 255.

Typ BOOLEAN je implementovaný v 1 byte, který nabývá hodnoty 0 a 1 (což odpovídá konstantám FALSE a TRUE).

Vyjmenované typy jsou implementované v 1 byte.

Typ TEXT (pokud je mu přiřazený diskový soubor) vytváří textové soubory v standardním tvaru operačního systému CP/M.

5.5 Omezení velikosti

Maximální vnoření procedur a funkcí:	4
Maximální počet identifikátorů v programu:	255

5.6 Vstup a výstup

Vstup a výstup je realizovaný pomocí proměnných typu TEXT stejně jako ve standardním Pascalu s těmito rozdíly:

Neexistují procedury GET a PUT.

Soubory použité v programu se nevyjmenovávají v hlavičce programu (jejich zapsání do hlavičky např. PROGRAM MOJ(INPUT,OUTPUT) způsobí chybu při překladu).

Před prvním použitím RESET nebo REWRITE je potřebné přiřadit souboru jméno pomocí procedury ASSIGN.

Mimo standardních souborů INPUT a OUTPUT, které jsou na začátku programu otevřené na klávesnici resp. obrazovku, existuje ještě soubor LST. Je na začátku programu otevřený na tiskárnu. Přiřazení souborům INPUT a OUTPUT je možné

změnit použitím ASSIGN a RESET resp. REWRITE, soubor LST není možné přesměrovat.

5.6.1 Vstup

Pro vstup ze souborů (z proměnných typu TEXT) slouží příkazy READ a READLN. Když je jejich prvním parametrem soubor, budou číst z něho. Když není, budou číst ze souboru INPUT. Soubor INPUT je na začátku programu přiřazený na vstup z klávesnice. Když nechceme toto přiřazení měnit, netřeba pro soubor INPUT použít ASSIGN ani RESET. Pomocí těchto procedur můžeme však přesměrovat soubor INPUT jinam.

READLN a EOLN funguje analogicky jako ve standardním Pascalu: když použijeme READ, i když platí EOLN, přečteme znak mezera, při dalším použití READ se bude žádat nový řádek z klávesnice. Situaci EOF je možné dosáhnout stlačením klávesy se šipkou vlevo s čárkou (šipka nahoru) - zobrazí se znak malý čtvereček (nedefinovaný znak) a potom EOL.

Příkazem READ můžeme číst proměnné celočíselné, reálné, znakové a řetězcové (typu ARRAY[1..N] OF CHAR, kde N je libovolné).

Pro čtení z klávesnice bez zobrazování znaků slouží funkce READKEY typu CHAR. Čeká na stlačení klávesy.

Logická funkce KEYPRESSED informuje o tom, zda je právě nějaká klávesa stlačena. Když je, tak mimo oznámení výsledku TRUE uloží stlačenou klávesu do vyrovnávací paměti, odkud si ji můžeme vyzdvihnout voláním READKEY - v tomto případě již READKEY nebude čekat na stlačení klávesy.

5.6.2 Výstup

Pro výstup slouží příkazy WRITE a WRITELN. Můžeme s nimi vypisovat celá a reálná čísla, znaky, řetězce a logické hodnoty (objekty typu Boolean).

Když je prvním parametrem příkazu WRITE nebo WRITELN soubor (proměnná typu TEXT), zápis bude vykonán do tohoto souboru. Před použitím WRITE však musí být tento soubor pojmenován procedurou ASSIGN a otevřen procedurou REWRITE. Když prvním parametrem WRITE nebo WRITELN není soubor, zápis bude vykonán do souboru OUTPUT.

Soubor OUTPUT je běžně nastaven na obrazovku - znaky se vypisují od pozice kurzoru, příkaz WRITELN způsobí přechod na nový řádek. Pozici kurzoru je možné změnit voláním procedury GOTOXY. Když nechceme toto přiřazení změnit, není třeba pro soubor OUTPUT použít ASSIGN ani REWRITE. Pomocí těchto procedur můžeme však přesměrovat soubor OUTPUT jinam.

5.6.2.1 Formátování výstupu

Za výrazem v příkazu WRITE nebo WRITELN může následovat dvojtečka a za ní celočíselný výraz. Tento výraz nazýváme formát. Například v příkazu WRITELN(X:8) je uvedený formát 8. Při výpisu reálných čísel můžeme použít také dva formáty, například WRITE(Y:15:6). První formát udává počet znaků, které má ve výpisu zabírat příslušná hodnota. Druhý formát u reálných čísel znamená počet míst za desetinnou tečkou. Když je 0, tak se číslo vypisuje v semilogaritmickém tvaru. V každém případě se z reálného čísla vypíše nejvíc 6 platných číslic, ostatní jsou nuly. Když je druhý formát větší než 38, přiřadí se mu hodnota 38.

Když je vypisovaná hodnota kratší než první formát, doplní se mezerami zleva. Když se číslo (celé nebo reálné) nedá vypsát na daný počet znaků, hodnota formátu se ignoruje a číslo se vypíše na nejmenší možný počet znaků. Když je řetězec kratší než první formát, usekne se.

Když v příkazu WRITE nebo WRITELN není uveden formát, doplní se standardní hodnota podle typu výrazu:

Typ výrazu	St. hodnota 1. formátu	St. hodnota 2. formátu
znak	1	
řetězec	délka řetězce	
logická hodnota	6	
celé číslo	6	
reálné číslo	12	0

5.7 Seznam standardních funkcí a procedur

Funkce a procedury, které nejsou v standardním Pascalu, jsou označeny hvězdičkou.

V seznamu používáme při popisu typu, parametru a výsledku tyto zvláštní typy:

SKALÁR libovolný skalární typ (INTEGER, CHAR, BOOLEAN
nebo vyjmenovaný typ)
ŘETĚZEC libovolný typ ARRAY [1..N] OF CHAR
UKAZATEL libovolný typ ukazatel

FUNCTION CHR(K: INTEGER): CHAR;
převod kód -> znak.

FUNCTION ORD(S: SKALÁR): INTEGER;
převod skalár -> pořadové číslo, speciálně pro znak je
to kód znaku.

FUNCTION SUCC(S: SKALÁR): SKALÁR;
nasledovník.

FUNCTION PRED(S: SKALÁR): SKALÁR;
předchůdce.

FUNCTION RAND(N: INTEGER): INTEGER; *

náhodné celé číslo od 0 do N-1.

FUNCTION ABS(I: INTEGER): INTEGER;
absolutní hodnota.

FUNCTION SQR(I: INTEGER): INTEGER;
druhá mocnina.

FUNCTION ODD(I: INTEGER): BOOLEAN;
výsledek je TRUE, když je I liché.

FUNCTION EOLN: BOOLEAN;
FUNCTION EOLN(VAR F: TEXT): BOOLEAN;
výsledek je TRUE, když je v souboru F zjištěný konec
řádku. EOLN bez parametru je ekvivalentní EOLN(INPUT).

FUNCTION EOF: BOOLEAN;
FUNCTION EOF(VAR F: TEXT): BOOLEAN;
výsledek je TRUE, když je na vstupu znak "konec
souboru". Znak konec souboru z klávesnice dosahneme
zmáčknutím klávesy se šipkou vlevo s čárkou (šipka

nahoru). EOF bez parametru je ekvivalentní EOF(INPUT).

FUNCTION KEYPRESSED:BOOLEAN;

*

výsledek je TRUE, když je v čase volání stisknuta nějaká klávesa na klávesnici. Výsledek je TRUE také tehdy, když byla stisknuta klávesa během některého předcházejícího volání funkce KEYPRESSED, ale mezitím nebyla volána funkce READKEY.

FUNCTION READKEY:CHAR;

*

výsledek je klávesa, která je stisknuta během volání funkce READKEY, nebo během některého předchozího vyvolání funkce KEYPRESSED. Když není žádná klávesa stisknuta, funkce čeká na její stisknutí. Stisknutá klávesa se nezobrazí na obrazovce.

PROCEDURE NEW(VAR S:UKAZATEL);

v paměti se přidělí místo pro jeden objekt typu, na který ukazuje ukazatel a do parametru se uloží odkaz na toto místo. Na rozdíl od standardního Pascalu, není možné volat tuto proceduru s více parametry.

PROCEDURE DISPOSE(S:UKAZATEL);

místo v paměti, na které ukazuje S se vyhlásí za volné. Při následujícím volání NEW se toto místo může znovu přidělit.

PROCEDURE MARK(VAR S:UKAZATEL);

do S se vloží ukazatel na konec volné paměti. Tedy adresa místa, které by se přidělilo při volání NEW. MARK se používá spolu s RELEASE.

PROCEDURE RELEASE(S:UKAZATEL);

nastaví začátek volné paměti na hodnotu, která je v S. Tím se vlastně uvolní všechna paměť, která byla přidělena procedurou NEW později, než bylo volané MARK(S). Nikdy nepoužíváme MARK a RELEASE dohromady s DISPOSE. RELEASE a DISPOSE reprezentují dva různé přístupy k uvolňování paměti a jejich společné používání může vést k problémům.

FUNCTION ADR(VAR X:LIBOVOLNÝ_TYP):INTEGER;

*

výsledkem funkce je adresa parametru.

FUNCTION INP(PORT:INTEGER):INTEGER;

*

výsledkem funkce je byte přečtený ze zadaného portu.

FUNCTION TRUNC(X:REAL):INTEGER;

převod čísla z REAL do INTEGER s useknutím desetinné části. Tedy $\text{TRUNC}(-7.8)=-7$, $\text{TRUNC}(5.4)=5$.

FUNCTION ROUND(X:REAL):INTEGER;

převod čísla z REAL do INTEGER se zaokrouhlením. Pro nezáporná čísla platí $\text{ROUND}(X)=\text{TRUNC}(X+0.5)$, pro záporná čísla platí $\text{ROUND}(X)=\text{TRUNC}(X-0.5)$.

FUNCTION RAND:REAL;

při volání funkce RAND bez parametru je výsledkem nahodné číslo v intervalu (0,1). *

FUNCTION ABS(X:REAL):REAL;

absolutní hodnota.

FUNCTION SQR(X:REAL):REAL;

druhá mocnina čísla.

FUNCTION SQRT(X:REAL):REAL;

druhá odmocnina čísla.

FUNCTION SIN(X:REAL):REAL;

sinus úhlu v radiánech.

FUNCTION COS(X:REAL):REAL;

cosinus úhlu v radiánech.

FUNCTION ARCTAN(X:REAL):REAL;

arcustangens - výsledek je v radiánech.

FUNCTION EXP(X:REAL):REAL;

funkce "e na X".

FUNCTION LN(X:REAL):REAL;

přirozený logaritmus.

FUNCTION BDOSA(FUN,DE:INTEGER):INTEGER;

*

FUNCTION BDOSHL(FUN,DE:INTEGER):INTEGER;

*

obě funkce slouží na vyvolání jádra systému CP/M - BDOSu. Jejich použití předpokládá znalosti operačního systému CP/M. FUN je číslo funkce, které se před voláním umístí do registru C a DE je hodnota, která se umístí do registru BC. Výsledkem funkce BDOSA je obsah registru A po návratu z BDOSu, výsledkem funkce BDOSHL je obsah registru HL. Když nepotřebujeme výsledné obsahy registrů, můžeme použít proceduru BDOS (viz dále).


```

FUNCTION BIOSA(FUN,BC: INTEGER): INTEGER;          *
FUNCTION BIOSHL(FUN,BC: INTEGER): INTEGER;         *

```

obě funkce slouží na vyvolání nejnižší úrovně operačního systému - BIOSu. Její použití předpokládá znalosti operačního systému CP/M. FUN je pořadové číslo funkce: WBOOT je 0, CONST je 1 atd. BC je hodnota, která se uloží do registru DE před voláním. Výsledkem funkce BIOSA je obsah registru A po návratu z BIOSu, výsledkem funkce BIOSHL je obsah registru HL. Když nepotřebujeme výsledné obsahy registrů, můžeme použít proceduru BIOS (viz dále).

```

PROCEDURE ASSIGN(VAR F:TEXT; JMÉNO:ŘETĚZEC);      *

```

souboru F se přiřadí jméno JMÉNO. Mimo jmen souborů na discích podle konvencí operačního systému CP/M můžeme použít také speciální jména zařízení 'CON:' a 'LST:'. Před RESET a REWRITE je nutné použít ASSIGN.

```

PROCEDURE RESET(VAR F:TEXT);

```

soubor F se otevře pro vstup. Když je to diskový soubor (tedy pomocí ASSIGN mu bylo přiřazeno jméno souboru na disku), hledá se tento soubor na disku a když se nenajde, vznikne chyba. Ze souboru otevřeného pomocí RESET je možné pouze číst.

```

PROCEDURE REWRITE(VAR F:TEXT);

```

soubor F se otevře pro výstup. Když je to diskový soubor (tedy pomocí ASSIGN mu bylo přiřazeno jméno souboru na disku), tak se na disku vytvoří nový soubor daného jména a případný starý soubor stejného jména se zruší. Do souboru otevřeného pomocí REWRITE je možné pouze zapisovat.

```

PROCEDURE CLOSE(VAR F:TEXT);                      *

```

uzavření souboru. Před novým RESET a REWRITE a před skončením programu je nutné použít pro výstupní soubory CLOSE, jinak není zaručené, že je v nich zapsáno všechno, co bylo vysláno příkazy WRITE. Vstupní soubory není nutné zavírat, ale doporučujeme to dělat.

```

PROCEDURE PAGE;
PROCEDURE PAGE(VAR F:TEXT);

```

vypíše kod "posuň na novou stranu". Je ekvivalentní WRITE(F,CHR(12)). PAGE bez parametru je ekvivalentní PAGE(OUTPUT). Na CON: se dosáhne smazání obrazovky, na tiskárně posunutí papíru na novou stranu

PROCEDURE PEN(I: INTEGER); *

nastavení barvy písmen, pozadí a způsobu kreslení (stejně jako v BASICU).

PROCEDURE GOTOXY(X,Y: INTEGER); *

nastavení kurzoru zařízení CON: na X-ty sloupec a Y-ty řádek.

PROCEDURE IPLOT(X,Y: INTEGER); *

nakreslí tečku na souřadnici X,Y. a přesune graficky kurzor na souřadnici X,Y. Souřadnice X je v intervalu 0 až 383, souřadnice Y v intervalu 0 až 255.

PROCEDURE BPLOT(X,Y: INTEGER; OBR: ŘETĚZEC); *

vykreslí obrázek přímo do paměti obrazovky - je to obdoba příkazu BPLOT v Basicu. X je od 0 do 47, Y od 0 do 255 (na rozdíl od Iplot se počítá shora). První byte parametru OBR musí obsahovat počet bajtů obrázku, druhý byte určuje, kolik byte se kreslí vedle sebe (tedy šířku obrázku v osmicích bodů), zbyte byty se vykreslí na obrazovku operací XOR. Byty se kreslí shora dolů a zleva doprava. Například
BPLOT(0,0,#8#2#\$AA#\$AA#\$55#\$55#\$AA#\$AA#\$55#\$55)
nakreslí šachovnici šířky 16 bodů (2x8) a výšky 4 body v levém horním rohu obrazovky.

PROCEDURE OUT(PORT,BYTE: INTEGER); *

vyšle hodnotu BYTE (v intervale 0..255) na port FORT.

PROCEDURE USR(ADRESA: INTEGER; VAR A,HL: INTEGER); *

provede se podprogram v strojovém kódu (většinou z monitoru C-2717) se zadanou adresou. Před voláním se do registrů A a HL umístí hodnoty parametrů A a HL. Po návratu se do parametry A uloží obsah registru A, do proměnné HL obsah páru HL (z parametru A se využívá jen nižší byte).

PROCEDURE BDOS(FUN,DE: INTEGER); *

procedura slouží na vyvolání jádra systému CP/M - BDOSu. Její použití předpokládá znalosti operačního systému CP/M. FUN je číslo funkce, které se před voláním umístí do registru C a DE je hodnota, která se umístí do registru DE. Procedura BDOS neumožňuje získat výsledné hodnoty registrů, na to slouží funkce BDOSA a BDOSHL (viz výše).

PROCEDURE BIOS(FUN,BC:INTEGER);

*

procedura slouží na vyvolání nejnižší úrovně operačního systému - BIOSu. Její použití předpokládá znalosti operačního systému CP/M. FUN je pořadové číslo funkce: WBOOT je 0, CONST je 1 atd. BC je hodnota, která se uloží do registru BC před voláním. Procedura BIOS neumožňuje získat výsledné hodnoty registrů, na to slouží funkce BIOSA a BIOSHL (viz vyše)

5.8 Zařízení

Uvedená jména můžeme použít místo jména souboru v proceduře ASSIGN (například ASSIGN(TISK,'LST:') přesměruje výstup na tiskárnu).

CON: Konzola. Vstupní i výstupní zařízení. Znaky se zobrazují na obrazovku, znak CHR(13) způsobí přechod na nový řádek. Znak CHR(12) způsobí vymazání obrazovky. Znak CHR(8) způsobí posun o jeden znak doleva. Příkazem READ je možné číst z klávesnice tak, že se vstupní písmena zobrazují na obrazovce tam, kde je nastavený kurzor. Během zápisu vstupu je zobrazován kurzor a je možné používat klávesy šipka vlevo nebo DEL na smazání špatných znaků. Když však vstup přechází přes předěl řádků, smazáním znaku se kurzor nevrátí na předešlý řádek (i když se znak v paměti smaže). Vstup je třeba ukončit stiskem klávesy EOL.

LST: Tiskárna. Výstupní zařízení.

Pascal C-2717 zná tři standardní soubory: INPUT, OUTPUT a LST. Na začátku běhu programu jsou INPUT i OUTPUT přiřazeny na zařízení CON: a LST:. Přiřazení souborů INPUT a OUTPUT můžeme během běhu programu měnit použitím ASSIGN, RESET a REWRITE. Přiřazení LST: nemůžeme už za běhu měnit.

1. Spuštění prekladače	2
2. Hlavní menu	2
3. Editor	3
3.1. Funkce kláves v editoru	3
4. Chybovník	4
5. Jazyk Pascal C-2717	8
5.1 Omezení proti standardnímu Pascalu	8
5.2 Rozšíření jazyka	8
5.3 Změny, které vyplývají z koncepce editoru	10
5.4 Implementace standardních typů	11
5.5 Omezení velikosti	11
5.6 Vstup a výstup	11
5.6.1 Vstup	12
5.6.2 Výstup	12
5.6.2.1 Formátování výstupu	13
5.7 Seznam standardních funkcí a procedur	14
5.8 Zařízení	19

05 / 512
574063

Název: Příručka programátora Pascalu C-2717 V2.C

Překlad: ing. Radek POKORNÝ /

Předloha: Návod na použití Pascalu C-2717 V2.00
RNDr. Peter TOMCSÁNYI

Vydal: INCOTEX Brno, Hybešova 42
ve spolupráci se s.p. Zbrojovka Brno, Lazaretní 7
a Stanicí mladých techniků Brno, Holubova 18

Cena: 10,- Kčs podle vyhlášky FCÚ č.35/1990 o smluvních cenách